A Robust Technique for Background Estimation in Heavily Intruded Videos

Bhumi Sabarwal, Priya Singh, and K S Venkatesh
Indian Institute of Technology/EE, Kanpur, India
Email: bhumi.s3006@gmail.com, {spriya, venkats}@iitk.ac.in

***Abstract*—The background estimation is often required in video surveillance applications. This paper presents some new techniques for background estimation in very high traffic conditions, where the background is visible for a very small fraction of the time. The algorithm exploits the fact that regions containing the background (assumed stationary) in the video sequence would always show a stable content, as opposed to the moving foreground which is highly transient, and presents an approach to eliminate the frames containing the foreground. Moreover, our proposed formulation of block wise stability provides a better estimate than a pixel wise approach. Results show that the obtained background comes very close to the actual background.**

***Index Terms*—background estimation, requantization, spatial consistency, temporal consistency**

I. Introduction

This Video surveillance systems aim to identify people, objects or any events of interest occurring in different environments. These systems consist of a module which performs background subtraction, for separating background pixels, which need to be ignored. Tin all such situations, a background model is required before this subtraction can be done. Background estimation has several problems explained in [1]. In this paper, we concentrate on the Bootstrapping problem, in which a training period absent of foreground objects is not available. This is a common scenario in traffic surveillance applications, especially when the traffic density is high as shown in Fig. 1. Hence we require an automatic background estimation algorithm, to estimate the background in such situations, even though it has never been available as such.

Figure 1. A sample frame of the background occluded by heavy traffic.

Manuscript received January 15, 2014; revised May 13, 2014.

II. Previous Work and Our Contribution for Paper Submission

A previous approach to background estimation was to use pixel-based average filter over a large number of frames. Zheng *et al.* [2] employed all incoming gray values of a pixel (including background and foreground object) to construct the Gaussian mod-el. However this algorithm is applicable for very low traffic and the resultant back-ground is very easily biased towards the foreground in case of high traffic. Kumar *et al.* [3] utilized a method to monitor the gray values from several frames without any foreground object for a few seconds. But in heavy traffic conditions it can be difficult to find enough foreground-free frames to build a reliable distribution of the back-ground image. Another way is to apply a median filter instead of aver-aging filter as illustrated in [4], but this also requires the background to be seen in at least in 50% of the cases. Hence all these algorithms are not applicable to background estimation where the traffic density is high, about 90%. In recent years Gaussian mixture model (GMM) based approaches to obtaining reliable background images have been developed [5], [6]. GMM-based methods feature effective background estimation under environmental variations through a mixture of Gaussians for each pixel in an image frame. However, this approach has an important shortcoming when applied to vision-based traffic monitoring systems (VTMS). For urban traffic, vehicles will stop occasionally at intersections because of traffic light. Such kind of transient stops will increase the weight of non-background Gaussian and seriously degrade the background estimation quality of a traffic image sequence. In such cases, we follow a different approach. Firstly, we seek to separate the input frames into foreground and back-ground regions. Periods of background content are identified by searching for the subset of frames with stable background content. The background, though available in only 10% cases, always shows similar and stable values. Finally, the background is estimated by applying the averaging filter over the background containing frames. A similar approach was introduced in [7]. It forms a block similarity matrix by plotting the difference of block values at different instants of time. It then uses this matrix to classify the frames into

doi: 10.12720/joig.2.1.46-53

foreground and background, by minimizing a cost function. However, this technique is quite complex. It recursively minimizes the cost function, which is time consuming. Moreover, it begins with an assumption that the pixel at the topmost left corner always contains the background, which is not valid in all cases. A much simpler and lesser computational approach to use the stability of the background is to plot pixel-wise histograms with the given data frames, as explained in [8]-[10]. The gray value with maximum frequency of occurrence is assigned as the background. This approach is robust to transient vehicle stops at road intersections, since, in the histogram, the weight due to this vehicle stop would be lesser than that obtained due to the stationary background for a decent amount of data. However, due to noise, even our stationary background may have a slightly different gray value every time. Moreover, it is possible that more than one gray value may have the same frequency of occurrence. An approach to overcome these two problems is illustrated in [11], where it forms Group-Based Histograms. In the Group-based histogram, a group based frequency is assigned to each gray value. This is an accumulative frequency, generated from its own frequency and the frequencies of the neighboring levels. The gray value corresponding to the highest group based frequency is assigned as the background. However, there is a shortcoming, if we have homogeneously colored traffic, the algorithm may give an erroneous result. In our work, we propose a new and yet simpler method of modifying the histograms to overcome the above problems. In Section 3 we propose to plot block-wise histograms which are more accurate than pixel-wise and discuss the consequences. We then propose to Re-quantize our data, which is a major contribution of our work. Instead of our 8-bit data, we re-quantize our data to be represented in 4 or 5 bits. We then plot histograms for our re-quantized data. This Re-quantized histogram will have lesser gray levels than 8-bit histograms, and thus have lesser computations. Also, it takes care of noise, as the adjacent gray levels due to the same background are merged into one value and contribute to the same peak. The Consistency Algorithm introduced takes care of the condition when many vehicles are of the same color.

Figure 2. The background estimate from median filtering.

Figure 3. The background obtained by our algorithm: in this example, we use pixel wise 4-bit requantization on 600 frames.

III. Background Reconstruction Algorithm

Our approach towards background reconstruction (Fig. 3), is applicable in very high density traffic conditions (and will of course also work in low density traffic too). The algorithm is based on the fact that the background, though available in very few frames, always shows a stable content, as compared to the continuously varying foreground.

A. Pixel Wise Algorithm

First, we work on gray scale images. We plot a histogram for each pixel position. This histogram is a plot of the range of gray values (0-255 in our 8-bit images), as base values of the histogram, versus the frequency of occurrence of these base values for the respective pixel position in the entire video. Initially, for the first few frames, we don't get a dominant peak, but gradually, over a large number of frames, we get a noticeable peak as seen in Fig. 4. The exact number of frames required is completely dependent on the traffic density and the environmental conditions. We look for the histogram peak; the corresponding base value is assigned as the estimated background value for the particular pixel position. The reason behind choosing the peak is that the background remains stable and every time it is seen, it contributes to the same base value in the histogram, unlike the highly varying foreground. The unstable foreground contributes towards different base values every time, and hence in the long run, the peak due to stable background dominates over all the other smaller peaks due to the unstable foreground. Indeed, this is not altogether new: this is the principle that underlies any bootstrapping estimation process. The pixel wise algorithm applied to 600 frames is shown in Fig. 5.

B. Block Wise Approach

The above algorithm utilizes pixel-level stability of the background. It has its shortcomings, however. There is a finite probability that a set of entirely foreground pixels at a location frequently assume the same value (sample space has 256 points). Should this happen, they will incorrectly pass for the background by causing a false peak in the histogram. A better approach in this regard would be to look for block wise stability of the

background. Applied to a block of N pixels, this probability of misrepresentation decreases to the Nth power of that of the pixel-wise false peak event. Hence such frames that contributed towards a false peak in pixel wise algorithm would never contribute to a false peak in the block wise approach. On the other hand, the resulting sample space has 256^N distinct events, so the forming of a representative histogram requires much more data.

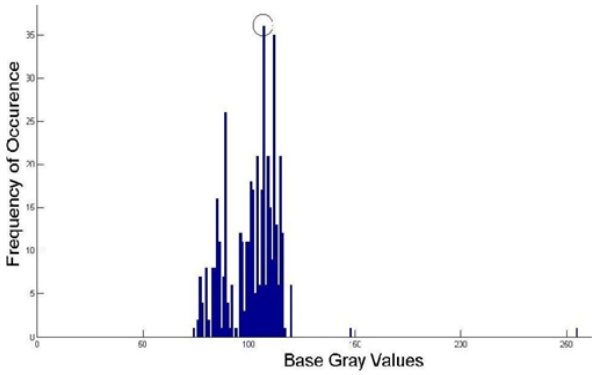

Figure 4. The Histogram plot for a pixel. The base value corresponding to the peak is assigned as background.

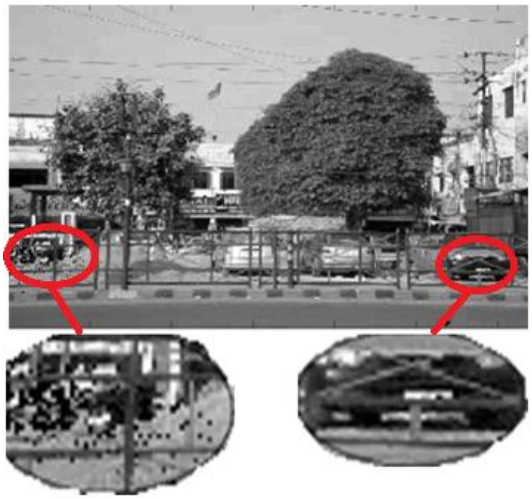

Figure 5. The pixel wise algorithm applied to 600 frames. A few regions where the algorithm is unable to estimate the background is shown enlarged.

Figure 6. The highly occluded surveillance image.

Figure 7. The estimated background obtained by pixel wise analysis on 600 frames.

C. Block Wise Algorithm

First we start with 2*1 blocks. Now, each block can have $2^8 \times 2^8$ values. The histogram is plotted combined for the 2 pixels in the block. It is a plot of 65536 'base values' (0, 0 to 255, 255), versus the frequency of occurrence of these values for each respective block position in all the frames. Then, as before, we find the peak and assign the corresponding gray values (a 2-tuple in this case) as the estimated background of the respective block position. Similarly, for a 2*2 block, the histogram would be plotted for $2^8 \times 2^8 \times 2^8 \times 2^8 = 2^{32}$ base values. However, plotting a histogram with these many base values is practically infeasible. Also, noise is a major factor, on which we have not focused yet. Considering both the above issues, our algorithms require some modification, to become both feasible and effective. We will be discussing the solution to both these problems in the coming sections.

IV. Requantization

As we saw in the previous section, implementing the 2*2 block wise algorithm is computationally expensive, because of the large number of bins (2^{32}) of the histogram. As a solution to this, we propose to re-quantize all the input images. Suppose that, instead of 256 grey values (8 bit images), we re-quantize the images to have just 16 grey values (4 bit images). The way to do this is to simply drop the 4 least significant bits at each pixel. With this 'requantization' the histogram would have only $2^4 \times 2^4 \times 2^4 \times 2^4 = 2^{16}$ base values for the 2*2 blocks in the block wise algorithm. This is computationally easier than before.

A. Block Wise Algorithm with Requantization

We follow the same approach as before and plot the histograms for the re-quantized images: therefore, for each block position, we have 2^{16} base values (as discussed), plotted versus the frequency of occurrence of these 4-tuple values for the respective block positions in

all the frames. Once we have the peak in the histogram for a particular block position, we separate out those frames which gave that peak. These frames are supposed to contain the stable background information for that particular block position. Hence we have separated the frames exclusively containing the relevant background from the rest. Now, to estimate the exact background value for each pixel position, we collect the 8-bit gray values of only those frames which were isolated as background carriers in the above exercise. The 8-bit gray value of the background is given by the weighted mean of all these values. Note that all the 4 pixels in a block would together contribute a common set of frames for estimating the background. As shown in Fig. 7 and Fig. 8, both pixel wise and block wise algorithms appear to work well on the highly occluded surveillance image shown in Fig. 6. However when we compared both in terms of Sum Of Absolute Differences from the actual background which we obtained by manually combining a careful selection of frames, the block wise algorithm proves to be significantly better.

Figure 8. The estimated background obtained by block wise 4 bit requantization algorithm on 600 frames using 2*2 blocks.

B. Noise Immunity Effects of Requantization

In section 3.1, we assumed that the stable and stationary background would always have the same gray value. However this is true only for a noise-free system. With noise present, it is not necessary that the background have zero sample variance over the available samples. In its present form, our algorithm would simply fail to find any background frames in such a situation. Hence, we ought to be looking at the neighborhood of the peak for the background, not solely at the peak itself. This is essentially what we are doing anyway under requantization. For example, in the above case, after re-quantizing all frames to 4 bits; 110, 108 and 112 would all be re-quantized to the same common value and hence would contribute to the same peak in the requantized histogram. The exact value of the background would be given as the weighted mean of all these values which contribute to the peak. Another advantage of requantization is that it requires less data to predict the background close to the actual than that without requantization. For example, in the above case, where the same background showed values of 110, 108 and 112; we would require a greater number of frames to highlight the dominant peak out of these. With requantization, they all would contribute to the same peak, and therefore, we would require fewer frames to get the dominant peak. We can simply assign the weighted mean of all these values as the estimated background. Fig. 9 & Fig. 10 show the improvement in the estimated background obtained by pixel wise requantization.

C. Requantization in Block-Wise Models

Requantization, which we found highly desirable even for the pixel-wise approach, now becomes indispensable when we model the background in blocks. In Section 3.2, we discussed background estimation by using a block wise algorithm with blocks of size 2*1.. Again, this approach that collects identical blocks is possible only in an ideal noiseless case. In a practical case, due to noise, it is almost impossible for a pair of background blocks to have exactly the same values as they had at some other instant. In the context of block-wise statistical modeling, requantization also brings down significantly the bin count of the histogram, making the construction of the histogram feasible.

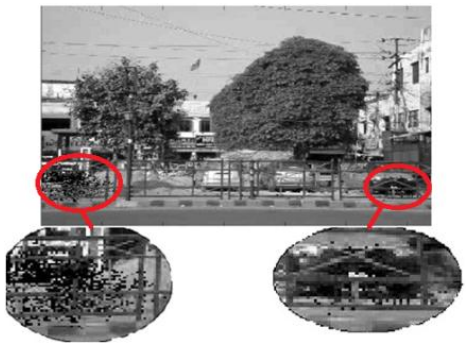

Figure 9. The estimated background obtained by pixel wise modeling, without requantization on 200 frames. There are many pixels where the algorithm fails to predict the background.

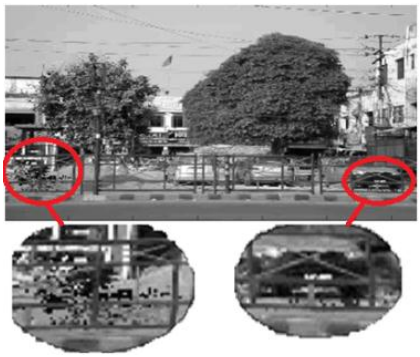

Figure 10. The estimated background obtained by pixel wise 4 bit requantization on 200 frames. It gives much better results than when applied without requantization.

V. Limitations of the Block Wise Algorithm

It may be concluded from our results that, we get more accurate background estimates from the block wise algorithms as compared to their pixel wise counterparts.

Moreover, block comparison with a 2*2 block size proves better than that with a 2*1 block size. One might argue that we can continue to increase the block size indefinitely to get still better results. However there is a limitation to the block sizes we can use. To understand this, we compare the pixel wise and block wise approaches (the latter with blocks of size 2*1). We have already mentioned that block approaches require longer samples to get reliable statistics out, as the entire pair of values in the block needs to occur sufficiently many times to contribute to the same peak as compared to the pixel wise case. For the same reasons, even larger strings of frames are necessary for 2*2 block sizes, since a 4-tuple of values needs to occur sufficiently many times. This leads to a practical problem: the background conditions tend to change gradually over a time span of minutes, usually due to illumination or other atmospheric changes. The background model's very validity is thus questionable if acquired over longer periods of time. This in turn sets a natural limit on the largest number of frames that may be gathered of any scene which is inherently non stationary—unless the frame rate is itself increased. For example, with blocks of size 4*4, the duration of the required string of frames (at 30fps) exceeds the stationarity limits of the scene background. Ultimately, this enforces a limit upon block size. Also, the block wise approach might not always be better than pixel wise algorithm. In our work, we found out that with microscopically non-static backgrounds, like those containing trees, where the leaves are continuously in motion, the pixel wise approach gives better results, since in this scenario it is more difficult that a block has exactly the same values at two time instants, than for a single pixel to do so. To handle this problem, we may ultimately need to make the block size space-varying and data-adaptive.

VI. Background Estimation in Colored Videos

Until now, we dealt only with gray scale images. We now see how to exploit availability of color information to further improve performance. A colored image has three values, R, G and B (each can have a value of 0 to 255). Therefore we need to look for background stability across all these three channels.

A. Background Estimation in Colored Videos

In order to look for background stability across these 3 separate channels, we propose to re-quantize all the three channels of the colored images. We then plot 3 separate pixel-wise histograms corresponding to each channel of R, G and B for each pixel position. Next we look at the peak of the histograms. We will have 3 different sets of frames contributing to the respective histogram peaks of each channel. We take the intersection of these three sets of frames and only consider these common frames for the estimation. For each channel, the estimated 8-bit background value is given by the weighted mean of all the respective 8-bit values in the respective channel in the common set of background frames for the pixel in question. Since we look for background stability across 3 separate channels, this gives us more accurate results than could be obtained from any one of the channels alone, or from the gray scale data. Note that requantization is a necessity while finding out the common frames through intersection of the 3 sets of frames. Without requantization, noise will affect the 3 channels differently and the histogram peaks might end up being contributed by 3 completely disjoint sets of frames: we might end up with no common frames at all to construct the background estimate with. Fig. 11 & Fig. 12 show the estimated background obtained for colored videos.

Figure 11. The estimated background in color using the color algorithm with 4 bit re-quantized images and 200 frames. The enlarged regions shows the pixels where no common frames are found across 3 channels of R, G, B.

Figure 12. The estimated background using the with 4 bit requantized color algorithm on 600 frames.

VII. The Time- Consistency Principle

In this section, we propose and exploit another property of a stable background. Whenever the background is seen in a video, it is likely to be seen *consistently* for a few frames in continuation, this is what we call the time-consistency property. Note the difference between stability and consistency. By the stability property of the background, we expect that every time the background is seen, it always has the same content. This

could happen in different, possibly temporally distant parts of the video. But in the case of time-consistency, we are concerned with one or more contiguous strings of frames of the video. To implement the time-consistency principle, we deal differently with time-consistent frames. Unlike the stability principle which assured only a proportionate representation for similar background frames in the histogram, our time consistency principle recommends that the representation of consecutively appearing background frames should be *more* than linearly proportionate to their number. We propose (tentatively) to square this frequency in case of time-consistent frames. Therefore, for 10 consistent background frames, the frequency of occurrence of that gray value would be treated as and not merely 10. As we saw in the earlier sections, noise always affects our algorithms. In a practical case, even though the background might be seen for 10 continuous frames, it is usual that that it has slight variations in its value within these 10 frames. Therefore, needless to say, we need to re-quantize our images. After requantization, the gray values of these 10 consistent frames correspond to nearly the same value and the same peak in the histogram.

A. *Time-Consistency Algorithm*

In our pixel wise time-consistency algorithm, we simply form a new data set which boosts time-consistent (consecutive) frames to the square of their actual number to get an artificially longer string of data. That is to say, every time we get a set of consistent frames for a pixel in the re-quantized images, we form a new data set for that pixel by simply repeating the entire string of the corresponding 8-bit values by as many times as the length of the string. Therefore, we have a new data set with longer strings of data – and therefore a greater overall length as well. Once we have formed the new data set, we follow the same procedure as set down in the earlier sections to estimate the background. In our work, we found out that consistency algorithm works well in videos with fast traffic even when the background is seen for very few frames in between. It suitably modifies the data set and we get a good estimate of the background. However, in slowly moving traffic or videos where vehicles stop in between, this algorithm doesn't work well, since momentary pauses in the traffic also give rise to undesirable manifestations of consistency.

VIII. RESULTS

We now compare the pixel wise approach, with and without the requantization algorithms. Also we see how requantizing to different number of bits changes the estimated background. The Fig. 13(a) show the estimated background obtained by pixel wise algorithm (without requantization) versus pixel wise with 3 bit, 4 bit and 5 bit requantization shown in Fig. 13 (b), (c) and (d) respectively. It is found that Pixel wise 4 bit Requantization Algorithm gives the best results for our data set as shown in Fig. 15. The choice of requantizing to appropriate bits depends on the data set noise levels.

(a)

(b)

(c)

(d)

Figure 13. (a) The estimated background obtained by pixel wise analysis, without requantization on 200 frames. (b) The estimated background obtained by pixel wise analysis with 3 bit requantization on 200 frames. (c) The estimated background obtained by pixel wise analysis with 4 bit requantization on 200 frames. (d) The estimated background obtained by pixel wise analysis using 5 bit requantization on 200 frames.

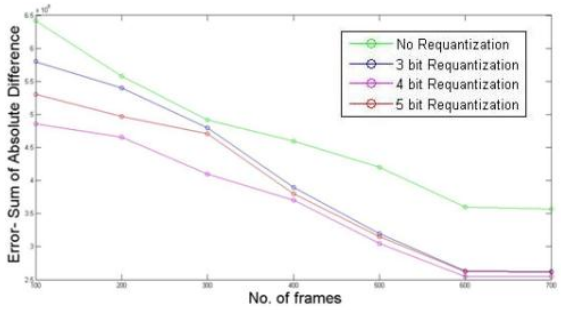

Figure 14. Performance comparison.

Figure 15. The estimated background obtained by pixel wise analysis with 4 bit requantization on 600 frames.

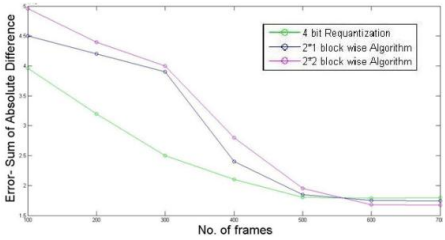

Figure 16. Performance comparison of the block wise and pixel wise approaches. Graph shows that after about 550 frames of data, block wise analysis begins to give better results than pixel wise analysis.

TABLE I. PERFORMANCE COMPARISON IN TERMS OF SUM OF ABSOLUTE DIFFERENCE FROM ACTUAL BACKGROUND. THE VIDEO FRAME IS SHOWN IN FIG. 1. IT SHOWS THAT FOR LESS DATA, THE COLOR ALGORITHM GIVES A HIGH ERROR, HOWEVER FOR LARGE DATA, THE RESULT IS MORE ACCURATE THAN OTHER ALGORITHMS. THE GBH AND PIXEL WISE REQUANTIZATION ALGORITHMS ALSO PERFORM WELL.

No. of frames	200	700
GBH Approach	$3.5*10^5$	$1.76*10^5$
4-bit req. pixel wise	$3.2*10^5$	$1.8*10^5$
2*1 block wise 4-bit req.	$4.2*10^5$	$1.75*10^5$
2*2 block wise 4-bit req.	$4.6*10^5$	$1.68*10^5$

TABLE II. PERFORMANCE COMPARISON IN TERMS OF SUM OF ABSOLUTE DIFFERENCE FROM ACTUAL BACKGROUND. WE TOOK A DIFFERENT VIDEO WHICH HAD STATIONARY BUILDINGS AND VEHICLES IN THE BACKGROUND, AS SHOWN IN FIG. 6, THEREFORE, THE SAD ARE IN A LOWER RANGE. FOR LESS DATA, PIXEL WISE ANALYSIS PERFORMS BETTER, HOWEVER FOR LARGE DATA, BLOCK WISE ANALYSIS IS BETTER.

No. of frames	500
GBH Approach	$15*10^5$
Pixel wise	$10*10^5$
4-bit req. temporal consistency	$7.2*10^5$

TABLE III. PERFORMANCE COMPARISON IN TERMS OF SUM OF ABSOLUTE DIFFERENCE FROM ACTUAL BACKGROUND. THE VIDEO FRAME IS SHOWN IN FIG. 13, THE FOREGROUND CONTAINED MANY WHITE COLORED VEHICLES. THE GBH ALGORITHM GAVE VERY HIGH ERROR. THE REQUANTIZATION ALGORITHM IS BETTER. HOWEVER THE TEMPORAL CONSISTENCY ALGORITHM OUTPERFORMS BOTH OF THEM.

No. of frames	200	700
GBH Approach	$5.2*10^5$	$2.75*10^5$
Pixel wise	$5.5*10^5$	$3.6*10^5$
4-bit req. pixel wise	$4.89*10^5$	$2.8*10^5$
4-bit req. pixel wise color	$5.3*10^5$	$2.65*10^5$

IX. CONCLUSION

We proposed many new algorithms for background estimation for heavy traffic videos. It must be clear at the outset that these approaches are meant for situations where the background is very scarce indeed. Not much work is available for handling such situations, hence comparative evaluation is difficult. Since such data is not common in standard datasets, we have had to shoot our own. This fact also explains our inability to demonstrate our proposed approaches on 'standard' datasets. As compared to previous algorithms that cannot estimate the background in such heavy traffic situations, our estimation approximates the background reasonably closely. In particular, our algorithm is more accurate than GBH (Kai-Tai Song, *et al.* 2008). We introduced requantization, to both combat noise and reduce computations and applied it with both the pixel wise and block wise models and got improved results, though at the cost of increased data length requirements, for the block wise approach. We showed how to exploit the presence of color. We introduced the time-consistency principle, which produced significantly superior estimates when traffic was in continuous motion but was problematic if the traffic became stationary.

REFERENCES

[1] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: Principles and practice of background maintenance," in *Proc. ICCV*, 1999, pp. 255.

[2] J. B. Zheng, D. D. Feng, W. C. Siu, Y. N. Zhang, X. Y. Wang, and R. C. Zhao, "The accurate extraction and tracking of moving objects for video surveillance," in *Proc. International Conference on Machine Learning and Cybernetics*, 2002, pp. 1909-1913.

[3] P. Kumar, K. Sengupta, and A. Lee, "A comparative study of different color spaces for foreground and shadow detection for traffic monitoring system," in *Proc. the 5th IEEE International Conference on Intelligent Transportation Systems*, 2002, pp. 100-105.

[4] M. Massey and W. Bender, "Salient stills: Process and practice," *IBM Systems Journal*, vol. 35, no. 3&4, pp. 557573, 1996.

[5] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 747-757, 2000.

[6] D. Butler, S. Sridharan, and V. M. Bove, "Real-time adaptive back-ground segmentation," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal*, 2003, pp. 349-352.

[7] D. Farin, P. H. N. De With, and W. Effelsberg, "Robust background estimation of complex video sequences," in *Proc. ICIP* 2003, 2003, pp. I-145-8 vol.1 .

[8] S. Kamijo, Y. Matsushita, K. Ikeuchi, and M. Sakauchi, "Traffic monitoring and accident detection at intersections," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, pp. 108-118, 2000.

[9] R. A. Johnson and G. K. Bhatacharyya, *Statistics: Principles and Methods*, John Wiley & Sons, New York, 2001.
[10] P. Kumar, S. Ranganath, W. Huang, and K. Sengupta, "Framework for real-time behavior interpretation from traffic video," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 1, pp. 43- 53, 2005.
[11] K. T. Song and J. C. Tai, "Real-time background estimation of traffic imagery using group-based histogram," *Journal of Information Science and Engineering*, vol. 24, pp. 411-423, 2008.

Bhumi Sabarwal received her M.Tech degree in Signal Processing from Indian Institute of Technology, Kanpur. Her area of interest includes Image Processing and Computer Vision.

Priya Singh received her B.E degree in Electronics and Instrumentation from RGPV University in 2011. Currently she is working as a Research Associate in Computer Vision lab at Indian Institute of Technology, Kanpur. Her area of interest includes Image Processing, Computer Vision and Robotics.

K.S. Venkatesh received his B.E degree in Electronics from Bangalore University, M.Tech degree in Communication and PhD in Signal Processing from Indian Institute of Technology, Kanpur. Currently, he is working as a professor at the Electrical engineering department in Indian Institute of Technology, Kanpur. His research interests include Image Processing, Video Processing, Computer Vision and Vision applications in Navigation and Robotics.